



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/042,079	01/07/2002	Scott J. Broussard	AUS920010996USI	6556
35525	7590	02/24/2005	EXAMINER	
IBM CORP (YA) C/O YEE & ASSOCIATES PC P.O. BOX 802333 DALLAS, TX 75380			ROMANO, JOHN J	
			ART UNIT	PAPER NUMBER
			2122	

DATE MAILED: 02/24/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary

Application No.

10/042,079

Applicant(s)

BROUSSARD, SCOTT J.

Examiner

John J Romano

Art Unit

2122

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 1/7/2002, 3/5/2002.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-39 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-39 is/are rejected.
- 7) ☒ Claim(s) 26 is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 05 March 0220 is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
 - ☐ Certified copies of the priority documents have been received in Application No. _____.
 - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- ☒ Notice of References Cited (PTO-892)
- ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date 3/5/2002.
- ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____
- ☐ Notice of Informal Patent Application (PTO-152)
- ☐ Other: _____

DETAILED ACTION

Claims 1-39 are pending in this action.

Information Disclosure Statement

1. The Information Disclosure Statement filed on March 5th, 2002 has been considered.

Claim Objections

2. Claim 26 objected to because of the following informalities: Claim reads according to claim 1, wherein this is an exact wording of claim 13. For compact prosecution, the examiner is deleting **[claim 1]** and inserting **claim 14** as seems appropriate. Appropriate correction is required.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

4. Claims **1, 2, 4-7, 9-15, 17-20, 22-28, 30-33 and 35-39** are rejected under 35 U.S.C. 103(a) as being unpatentable over Krishna et al., US 2003/0051233 A1 (hereinafter **Krishna**) in view of Dale Green, "Trail: The Reflection API", The Java

Tutorial. Posted November 27th, 1999. Retrieved from
<<http://Green.com/docs/books/tutorial/reflect/>> (hereinafter **Green**).

5. In regard to claim 1, **Krishna** discloses:

- *"A method in a data processing system for generating a generic compilation interface from a first object-oriented software package, said method comprising the steps of..."* (E.g., see Figure 7A & Page 2, Paragraph [0025]), wherein the library stubs exclude the source code executable statements, but include (generic), declarations and interfaces so that the secondary developer can compile class files for converting to CAP files, etc (interface).
- *"...removing all references to software that is defined in a second software package from said public entities included in each of said public classes..."* (E.g., see Figure 7B & Page 2, Paragraph [0025]), wherein the library stubs exclude (remove), the source code executable statements (software defined in a second package), but include declarations and interfaces so that the secondary developer can compile class files, wherein Figure 7B, steps 721- 734, teach replacing a reference returned with the appropriate return type value.
- *"...generating an equivalent public class for each of said identified public classes, said equivalent public class including equivalent public entities that include no references to said software defined in said*

second package..." (E.g., see Figure 7A & 7B, steps 737-747 & Page 3, Paragraph [0036]), wherein equivalent public class for each of said identified public class are generated, wherein "...only non-private (public) method signatures and field signatures are needed for off-card compiling and conversion...is sufficient for synthesizing the library stubs 220 (Figure 3)".

- *"...compiling each of said equivalent public classes; and generating a compilation interface for said first package including each of said compiled equivalent public classes."* (E.g., see Figure 7A & 7B & Page 3, Paragraph [0048]), wherein the pseudo code teaches generating (compiling) an equivalent JAR file (Step 747).

But **Krishna** does not expressly disclose "*...identifying all public classes included in said first software package ...*" or "*...for each of said public classes, identifying all public entities included in each of said public classes ...*". However, **Green** discloses:

- *"...identifying all public classes included in said first software package ..."* (E.g., see "Discovering Class Modifiers", Page 7), wherein all public class modifiers are discovered.
- *"...for each of said public classes, identifying all public entities included in each of said public classes ..."* (E.g., see "Trail: The Reflection API", Page 1, Paragraph 1), wherein all public class modifiers are discovered along with their fields, methods and variables (entities).

Krishna and **Green** are analogous art because they are both concerned with the same field of endeavor, namely, using the JAVA language to examine, manipulate and work with classes. Therefore, at the time the invention was made, it would have been obvious to a person of ordinary skill in the art to combine a public class modifiers and their attributes with **Krishna's** JAVA program for interpreting, interfacing and compiling. The motivation to do so, is suggested by **Krishna**, "...only non-private method signatures and field signatures are needed... for compiling...", (Page 3, Paragraph [0035]). Furthermore, **Green** suggests "...to use the reflection API if you are writing development tools..." (Page 1, Paragraph 1).

6. In regard to claim **2**, the rejections of base claim **1** are incorporated.

Furthermore, **Green** discloses:

- "...identifying all entities included in each of said public classes that include a public modifier." (E.g., see "Trail: The Reflection API", Page 1), wherein all public class modifiers are discovered along with their fields, methods, superclasses and variables (entities).

Therefore, at the time the invention was made, it would have been obvious to a person of ordinary skill in the art to combine a public class modifiers and their entities with **Krishna's** JAVA program for interpreting, interfacing and compiling. The motivation to do so, is suggested by **Krishna**, "...only non-private method signatures and field signatures are needed... for compiling...", (Page 3, Paragraph [0035]).

Furthermore, **Green** suggests "...to use the reflection API if you are writing development tools..." (E.g., see "Trail: The Reflection API", Page 1).

7. In regard to claim 4, the rejections of base claim 1 are incorporated.

Furthermore, **Green** discloses:

- "...*identifying all public methods included in each of said public classes.*" (E.g., see "Obtaining Method Information", Page 15), wherein one can "...uncover a method's name, return type, parameter types, set of modifiers, and set of throwable exceptions."

Therefore, at the time the invention was made, it would have been obvious to a person of ordinary skill in the art to combine public methods modifiers and their entities with **Krishna's** JAVA program for interpreting, interfacing and compiling. The motivation to do so, is suggested by **Krishna**, "...only non-private method signatures and field signatures are needed... for compiling...", (Page 3, Paragraph [0035]).

Furthermore, **Green** suggests "...to use the reflection API if you are writing development tools..." (E.g., see "Trail: The Reflection API", Page 1, Paragraph 1).

8. In regard to claim 5, the rejections of base claim 1 are incorporated.

Furthermore, **Green** discloses:

- "...*public parameters included in each of said public classes.*" (E.g., see "Obtaining Method Information", Page 15, Paragraph 2), wherein "... the following sample program prints the name, return type, and parameter types of every public method in the Polygon class", wherein the public parameters are included.

Therefore, at the time the invention was made, it would have been obvious to a person of ordinary skill in the art to combine public parameters with **Krishna's** JAVA program for interpreting, interfacing and compiling. The motivation to do so, is suggested by **Krishna**, "...only non-private method signatures and field signatures are needed... for compiling...", (Page 3, Paragraph [0035]). Furthermore, **Green** suggests "...to use the reflection API if you are writing development tools..." (E.g., see "Trail: The Reflection API", Page 1, Paragraph 1).

9. In regard to claim 6, the rejections of base claim 1 are incorporated.

Furthermore, **Green** discloses:

- "...*public fields included in each of said public classes.*" (E.g., see "Trail: The Reflection API", Page 1), wherein all public class modifiers are discovered along with their fields, methods and variables (entities).

Therefore, at the time the invention was made, it would have been obvious to a person of ordinary skill in the art to combine a public class modifiers and their fields with **Krishna's** JAVA program for interpreting, interfacing and compiling. The motivation to do so, is suggested by **Krishna**, "...only non-private method signatures and field signatures are needed... for compiling...", (Page 3, Paragraph [0035]). Furthermore, **Green** suggests "...get information about a class's ...fields..." (Page 1, Paragraph 1).

10. In regard to claim 7, the rejections of base claim 1 are incorporated.

Furthermore, **Krishna** discloses:

- "...*identifying all public classes included in a Java Archive file.*" (E.g., see Figure 7A, step 704), wherein an IDE file from which to synthesize classes/jar file is disclosed.

11. In regard to claim 9, the rejections of base claim 1 are incorporated.

Furthermore, **Green** discloses:

- "...*identifying all public entities included in each of said public classes utilizing Java Reflection.*" (E.g., see "Trail: The Reflection API", Page 1), wherein all public class modifiers are discovered along with their fields, methods and variables (entities) using the Java Reflection API.

Therefore, at the time the invention was made, it would have been obvious to a person of ordinary skill in the art to combine Java Reflection with **Krishna's** JAVA program for interpreting, interfacing and compiling. The motivation to do so, is suggested by **Krishna**, "...only non-private method signatures and field signatures are needed... for compiling...", (Page 3, Paragraph [0035]). Furthermore, **Green** suggests "...to use the reflection API if you are writing development tools..." (E.g., see "Trail: The Reflection API", Page 1, Paragraph 1).

12. In regard to claim 10, the rejections of base claim 1 are incorporated.

Furthermore, **Krishna** discloses:

- "...*generating a separate java file for each of said identified public classes.*" (E.g., see Figure 7B, step 738), wherein a separate .java file is generated for each of said identified public classes.

Art Unit: 2122

13. In regard to claim **11**, the rejections of base claim **10** are incorporated.

Furthermore, **Krishna** discloses:

- "...*compiling each said java file.*" (E.g., see Figure 7B, step 744), wherein a separate .java file is compiled.

14. In regard to claim **12**, the rejections of base claim **11** are incorporated.

Furthermore, **Krishna** discloses:

- "...*generating a compilation Java Archive file; and storing each said compiled .java file in said compilation Java Archive file.*" (E.g., see Figure 7B, step 747), wherein the class files are placed in a Java Archive file.

15. In regard to claim **13**, the rejections of base claim **1** are incorporated.

Furthermore, **Krishna** discloses:

- "...*utilizing said compilation interface within an Integrated Development Environment.*" (E.g., see Figure 1 and Paragraph [0005]), wherein a standard Java development environment is disclosed.

16. As per claims **14**, **15**, **17-20** and **22-26**, this is a system version of the claimed method discussed above, in claims **1,2**, **4-7** and **9-12**, wherein all claimed limitations have also been addressed and/or cited as set forth above. For example, see **Krishna** (Figure 8), wherein a system employing the above methods is disclosed.

17. As per claims **27**, **28**, **30-33** and **35-39**, this is a product version of the claimed method discussed above, in claims **1,2**, **4-7** and **9-12**, wherein all claimed limitations have also been addressed and/or cited as set forth above. For example, see **Krishna**

(Figure 1 & Paragraph [0053]), wherein a product version of the above methods is disclosed.

18. Claims **3**, **16** and **29** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Krishna** in view of **Green** and further in view of Evans et al., US 6,836,884 B1 (hereinafter **Evans**).

19. In regard to claim **3**, the rejections of base claim **1** are incorporated. But the combined teaching of **Krishna** and **Green** do not expressly disclose, "...*determining whether each of said entities includes a native attribute; in response to a determination that each of said entities includes a native attribute, removing said native attribute from each of said entities; and generating equivalent entities that include no native attributes.*" However, **Evans** discloses:

- "...*determining whether each of said entities includes a native attribute; in response to a determination that each of said entities includes a native attribute, removing said native attribute from each of said entities; and generating equivalent entities that include no native attributes.*" (E.g., see Figure 3 & Column 12, line 63 – Column 13, line 11), wherein native code may be edited from one form to a more general form.

Evans and the combined teaching of **Krishna** and **Green** are analogous art because they are both concerned with the same field of endeavor, namely, compiling software code. Therefore, at the time the invention was made, it would have been

obvious to a person of ordinary skill in the art to combine replacing native code with the combined teachings method. The motivation to do so, is suggested by **Evans**, "...the user may replace an existing method created in a first source language with a new method created in a second source language.", (E.g., see Column 2, line 65 – Column 3, line 1).

20. As per claim **16**, this is a system version of the claimed method discussed above, in claim **3**, wherein all claimed limitations have also been addressed and/or cited as set forth above. For example, see **Krishna** (Figure 8), wherein a system employing the above method is disclosed.

21. As per claim **29**, this is a product version of the claimed method discussed above, in claim **3**, wherein all claimed limitations have also been addressed and/or cited as set forth above. For example, see **Krishna** (Figure 1 & Paragraph [0053]), wherein a product version of the above method is disclosed.

22. Claims **8**, **21** and **34** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Krishna** in view of **Green** and further in view of obviousness.

23. In regard to claim **8**, the rejections of base claim **1** are incorporated. But the combined teaching of **Krishna** and **Green** do not expressly disclose, "...utilizing a *java.util.jar* utility." However, it would have been obvious, to one of ordinary skill in the art to utilize a *java.util.jar* utility to identify all public classes included in said first package. The motivation to do so is provided by **Green** (E.g., see "Examining Interfaces", Page 1), wherein importing *java.util.** is shown, wherein the *java.util.jar*

utility would be included. Furthermore, the java.util.jar is another known method to extract class information and thus it would have been obvious to one of ordinary skill in the art to use a utility already imported to achieve a goal already stated (identify all public classes). Therefore, at the time the invention was made it would have been obvious to identify all public classes included in said first package utilizing a java.util.jar utility with the combined teaching of **Krishna** and **Green**.

24. As per claim **21**, this is a system version of the claimed method discussed above, in claim **8**, wherein all claimed limitations have also been addressed and/or cited as set forth above. For example, see **Krishna** (Figure 8), wherein a system employing the above method is disclosed.

25. As per claim **34**, this is a product version of the claimed method discussed above, in claim **8**, wherein all claimed limitations have also been addressed and/or cited as set forth above. For example, see **Krishna** (Figure 1 & Paragraph [0053]), wherein a product version of the above methods is disclosed.

Conclusion

26. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

- Lewis et al., US005590331A
- Lee et al., US 20020147763A1


Art Unit: 2122

27. Any inquiry concerning this communication or earlier communications from the examiner should be directed to John J Romano whose telephone number is (571) 272-3872. The examiner can normally be reached on 8-5:30, M-F.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

JJR



TUAN DAM
SUPERVISORY PATENT EXAMINER